Section C: API Testing

Prepare a collection named “ZSS API Test_YOUR NAME” and an environment named “Env_YOUR NAME” in Postman. Create necessary folders or subfolders to manage the test scenario stated below. Prepare all necessary environment variables so that no values are hardcoded in the payload -

1. Create a POST request named “Login Token Generation” using following data set -

   Request Type : POST
   Header : application/json
   Endpoint : ***{{url}}/api-token-auth/gen/***
   URL: **https://nmed-c.zssbd.com/**
   Data to be passed :
   username : testdoc
   password : Test123456

Write necessary tests to capture the token generated from successful post request and save it to a variable to pass this token in the next API requests.

2. Create a **POST** request named “Prescription Create” using following data set

Prescriptions List/Create API:

- **<base_url>/api/prescriptions/**
  - GET/POST
  - Authentication **REQUIRED should be passed in header following - token {{token}} format**
  - Read/Write: **DOCTOR**
  - Query params: patient (patient id)

REQUEST DATA:

```
{
   "patient": <patient id>,
   "additional_notes": <additional_notes in string>,
   "prescribed_medicines": [
      {
         "medicine": <medicine_id>,
         "dosage_1": <dosage amount in integer>,
         "dosage_2": <dosage amount in integer>,
         "dosage_3": <dosage amount in integer>,
         "duration": <duration in integer>,
```

```
            "meal_reliance": <meal_reliance integer value>,
        }
    ],
    "medical_tests": [list of medical test id]
}
```

Requirements Are-

1. Either Additional notes OR Prescribed medicines OR Medical tests are necessary
2. Patient Id value is - 773
3. Fields of Prescribes medicines will be integer, meal reliance value will be 1 or 2
4. Medical tests field will be integer and will get from DB so range will be - 1 to 8

Here make folder for Prescription create and add all necessary test cases to check the validation of fields and requirements.

3. Create a **GET** request named “Prescription Get” with the end point
<base_url>/api/prescriptions/prescription_id/
(Authorization will be needed)
Write test scripts for retrieving the created prescription from the GET request and validate them

4. Create a **PUT** request named “Prescription Update” with the end point
<base_url>/api/prescriptions/prescription_id/
(Authorization will be needed)
Write test scripts for updating the created prescription and again validate the change using GET request.

5. Finally delete the created prescription using **DELETE** request named “Delete Prescription” with the end point
<base_url>/api/prescriptions/prescription_id/
(Authorization will be needed)
Write test scripts for validating prescription Deletion and again validate the change using GET request.

After completing the project, Share the collection with a public link and upload the Environment export file in the two separate boxes in the question.